WSDL, UDDI, .NET
March 30, 2005

Web Services Architecture
- Service Description Languages provide information about how to call a service. Typically, a schema of the request and a schema of the response is included.

Service Description
- SDL (Service Description Language) MS v1.0
- SCL (Service Contract Language) MS v2.0
- NASSL (Network Accessible Service Specification Language) IBM v1.0
- WSDL (Web Services Description Language)

WSDL
- Web Services Description Language
- WSDL is written in XML
- WSDL is used to describe Web services
  - what operations does the service expose?
  - plus the URL of service
  - needs to be distributed to clients
  - compile to create stubs
  - use stubs to access web service
- Problem: WSDL files are huge
  - 66 lines to describe ‘int GetStockPrice(char *symbol, float *Result);’
  - often WSDL specs generated out of Java/C interface instead of WSDL file

WSDL Components
- Types – defined in schema
- Message
- Operation – an abstract service
- Binding
- Port – operation ‘bound’ to a defined binding
- Port type and Service – collections

WSDL
- Types
  - Schema
- Operation
  - message
  - message
  - in
  - out
WSDL Components

- Port type
  - Operations
  - Bindings

WSDL Structure

```xml
<definitions>
  <types>
    definition of types...
  </types>
  <message>
    definition of a message...
  </message>
  <portType>
    definition of a port...
  </portType>
  <binding>
    definition of a binding...
  </binding>
  <service>
    definition of a service...
  </service>
</definitions>
```

WSDL Ports

- The `<portType>` element is the most important WSDL element.
- It defines a web service, the operations that can be performed, and the messages that are involved.
- The `<portType>` element can be compared to a function library (or a module, or a class) in a traditional programming language.

WSDL Messages

- The `<message>` element defines the data elements of an operation.
- Each messages can consist of one or more parts. The parts can be compared to the parameters of a function call in a traditional programming language.
WSDL Types

- The `<types>` element defines the data type that are used by the web service.
- For maximum platform neutrality, WSDL uses XML Schema syntax to define data types.

WSDL Operation Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-way</td>
<td>The operation can receive a message but will not return a response</td>
</tr>
<tr>
<td>Request-response</td>
<td>The operation can receive a request and will return a response</td>
</tr>
<tr>
<td>Solicit-response</td>
<td>The operation can send a request and will wait for a response</td>
</tr>
<tr>
<td>Notification</td>
<td>The operation can send a message but will not wait for a response</td>
</tr>
</tbody>
</table>

WSDL Sample Binding

```xml
<binding type="glossaryTerms" name="b1">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
  <operation>
    <soap:operation soapAction="http://example.com/getTerm"/>
    <input> <soap:body use="literal"/> </input>
    <output> <soap:body use="literal"/> </output>
  </operation>
</binding>
```

WSDL Bindings

- The `<binding>` element defines the message format and protocol details for each port.

Web Service Description

```xml
<?xml version="1.0">
<definitions name="StockQuote">
  <types>
    <schema>…</schema>
  </types>
  <message name="GetQuoteInput">
    <part name="symbol" type="xsd:string"/>
  </message>
  <message name="GetQuoteOutput">
    <part name="symbol" type="xsd:float"/>
  </message>
</definitions>
```

Web Service Description (2)

```xml
  <portType name="StockQuotePortType">
    <operation name="GetQuote">
      <input message="GetQuoteInput"/>
      <output message="GetQuoteOutput"/>
    </operation>
  </portType>
```
Web Service Description (3)

```xml
<binding name="StockQuoteSoapBinding"
  type="StockQuotePortType">
  <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="GetQuote">
    <soap:operation soapAction="/">
      <input>
        <soap:body namespace="http://example.com/stockquote.xsd" /></input>
      <output>
        <soap:body namespace="http://example.com/stockquote.xsd" /></output>
    </soap:operation>
  </operation>
</binding>
```

Web Service Description (4)

```xml
  <service name="StockQuoteService">
    <port name="StockQuotePort" binding="StockQuoteSoapBinding">
      <soap:address location="http://example.com/stockquote"/>
    </port>
  </service>
</definitions>
```

Discovery

- Various methods to find services exist
  - Manual: call/email
  - Look-up in a known source of service information (like a phonebook)
  - Peer to peer automated search

Registries

- A web services registry provides an interface to browse/query a directory of web service descriptions

  - Two specifications in this space:
    - UDDI (Universal Description, Discovery, and Integration)
      - http://www.uddi.org/
    - ebXML Registry and Repository
      - http://ebxml.org/project_teams/registry/registry.htm

Registries

1. Businesses populate the registry with descriptions of the services they support
2. The registry assigns a programmatically unique identifier to each registration and stores it
3. Marketplaces, search engines, and business apps query the registry to discover services at other companies

Business Registration

- Common Registration
  - Basic information a company registers about itself for all marketplaces
White Pages
- Business Name
- Text Description (list of multi-language text strings)
- Contact info (list of names, phone numbers, fax numbers, websites, etc)
- Known Identifiers (list of identifiers that a business may be known by - DUNS, Thomas, other)

Yellow Pages
- Business categories
- 3 standard taxonomies
- Industry (NAICS)
- Product/Services: (UN/SPSC)
- Location
- Implemented as name-value pairs to allow any valid taxonomy identifier to be attached to the business white page

Green Pages
- New set of information businesses use to describe how to “do e-commerce” with them
- Business processes
  - One or more service interfaces that support each process
  - Process specific contact information
  - Process description in multiple languages
  - Categorization of processes
  - Pointers to specifications describing each API

UDDI Registration
- XML document
- Created by end-user company (or on their behalf)
- Can have multiple service listings
- Can have multiple taxonomy listings

Example of Registration

Registry Operation
- Peer nodes (websites)
- Companies register with any node
- Registrations replicated on a daily basis
- Complete set of "registered" records available at all nodes
- Common set of SOAP APIs supported by all nodes
UDDI and SOAP

UDDI and SOAP

Registry Interface (SOAP Messages)

• Inquiry API
  - Find things
    - find_business
    - find_service
    - find_binding
    - find_tModel
  - Get Details about things
    - get_businessDetail
    - get_serviceDetail
    - get_bindingDetail
    - get_tModelDetail

• Publishers API
  - Save things
    - save_business
    - save_service
    - save_binding
    - save_tModel
  - Delete things
    - delete_business
    - delete_service
    - delete_binding
    - delete_tModel
  - Security
    - get_authToken
    - discard_authToken

UDDI Versions

• Version 1: 2000, most widely deployed
• Version 2: 2001
  - Service workflow
  - Custom categorizations
• Version 3: 2002
  - Subscriptions/notifications
  - Security

jUDDI

• Open-source implementation of UDDI
• Applicable to both registry and client-side but client-side is now focus
• Hosted by SourceForge

http://www.juddi.org/

XML-RPC vs SOAP

• XML-RPC is much simpler
• There are lots of web services based on XML-RPC.
• SOAP makes it easier to exchange more complex documents.
• SOAP runs over many protocols:
  - HTTP, SMTP, FTP, ...

Building Web Services

• Visual Studio .NET – developed by Microsoft
  http://www.microsoft.com/net/
• WebSphere – developed by IBM
• Sun Java Studio Creator – developed by Sun Microsystems
  http://www.sun.com/software/products/jscreator/
  (tool for building Web Application will be released in July 2004)
  (tool for building Web Services will be released in Dec 2004)
.NET

- Software as a service: access software from variety of platforms (handhelds to servers) and many operating systems
- Language neutral
- Component based
- Collection of bits of software, including SOAP

.NET Framework

- The .NET framework is the infrastructure for the overall .NET Platform.
- Major Components of the framework
  - Common Language Runtime (CLR)
  - Base Class Library
  - Common Type System (CTS)
  - Common Language Specification (CLS)

Framework Structure

Web Services in .NET

- Enables disparate systems to exchange data using internet standards (http, xml, soap)
- Pieces of program logic that are exposed publicly (UDDI)
- Black boxes of program logic that abstract the functionality away from consuming application
- Consuming application needs only to know how to invoke the web service and what to expect in return

3 Steps to Consuming Web Services

- Discovering XML Web Service
  - May use the Universal Description, Discovery, and Integration service (UDDI)
  - Result of discovery is the URL to the Web Service Description (WSDL) document
- Creating a Proxy
  - Proxy is a class on the local server which exposes the same interfaces as the XML Web Service
  - We call the Web Service to invoke the specified Web method.
  - Using .NET framework we can use the WSDL.exe utility which automatically generates a proxy class for the service
- Creating an instance and invoking
  - Once we have proxy class, we can create an instance and invoke any of it's methods
  - When method invoked, proxy calls out to XML web service and invokes same method
  - Proxy then exposes output to original caller, as if method were invoked locally

Consuming XML Web Services

- The consumer finds an available XML Web Service either in the UDDI registry, or directly from the provider (not shown).
- The consumer makes a request to the provider for the WSDL document.
- The provider receives the XML-based WSDL document.
- The consumer makes a request to the XML Web Service in the WSDL document, passing in any required arguments as defined in the WSDL.
- The provider processes the request, performing any functions necessary to complete the request, and returns the results to the consumer, either as either an XML document, or a SOAP message.
Building Web Services

- Must create a .asmx file which defines the class and methods of XML Web Service
- .asmx file does not define user interface
- Must declare classes as WebService, and methods as WebMethod

Simple Hello World Example

Hello World Web Service using C#

```csharp
<%@ WebService Language="c#" Class="MyService" %>
using System;
using System.Web.Services;

[WebService(Namespace="http://www.doeagent.com")]
public class MyService : WebService
{
    [WebMethod()]
    public string SayHello( String YourName )
    {
        return "Hello, " + YourName;
    }
}
```

JAX-RPC

- JAX-RPC -- Java API for XML-based RPC.
- Calls and responses are transmitted as SOAP messages over HTTP.
- A JAX-RPC client can access a Web service that is not running on the Java platform and vice versa.
- Hello World Example:  
  http://java.sun.com/webservices/docs/1.0/tutorial/doc/JAXRPC3.html

Mono

- Open-source implementation of .NET framework
- Runs on Linux, Windows, OSX, BSD, Netware
- http://www.mono-project.com